Fundamentos de Algoritmia

Grado en Ingeniería Informática. Grupos A,C y DG

Examen final, 14 de Enero de 2020.

Normas de realización del examen

1. Debes programar soluciones para cada uno de los tres ejercicios, probarlas y entregarlas en el juez automático accesible en la dirección `http://exacrc/domjudge/team`.
2. Escribe comentarios que expliquen tu solución, justifiquen por qué se ha hecho así y ayuden a entenderla. Calcula la complejidad de todas las funciones que implementes.
3. En el juez te identificarás con el nombre de usuario y contraseña que has recibido al comienzo del examen. El nombre de usuario y contraseña que has estado utilizando durante la evaluación continua **no** son válidos.
4. Escribe tu **nombre y apellidos** en un comentario en la primera línea de cada fichero que subas al juez.
5. Tus soluciones serán evaluadas por la profesora independientemente del veredicto del juez automático. Para ello, la profesora tendrá en cuenta **exclusivamente** el último envío que hayas realizado de cada ejercicio.
6. Al finalizar el examen no olvides **firmar y añadir la hora de entrega** en la hoja de firmas.

1. **(3.5 puntos)** Dado una secuencia de n enteros ($n \geq 0$) se desea encontrar la longitud del segmento más largo que no contiene ningún subsegmento estrictamente creciente de tres elementos.

   1. (0.75 puntos) *Especifica formalmente* una función que dado un vector de enteros de longitud ≥ 0 devuelva la longitud del segmento más largo que no contiene ningún subsegmento estrictamente creciente de tres elementos.
   2. (1.5 puntos) *Diseña e implementa* un algoritmo iterativo que resuelva el problema propuesto.
   3. (0.75 puntos) Escribe el *invariante* del bucle que permite demostrar la corrección del mismo y proporciona una función de cota.
   4. (0.5 puntos) Indica el *coste asintótico* del algoritmo en el caso peor y justifica adecuadamente tu respuesta.

Entrada

La entrada comienza con una línea que contiene el número de casos de prueba. Cada caso de prueba contendrá el valor del número de elementos n y a continuación los elementos de la secuencia.

Salida

Por cada caso de prueba el programa escribirá una línea con la longitud del segmento más largo solicitado en el enunciado.

Entrada de ejemplo

```
5
8
1 2 3 1 2 0 4 5
2
2 -1
5
1 1 1 2 2
6
0 4 1 2 -3 7
1
7
```

Salida de ejemplo

```
6
2
5
6
1
```

2.(2.5 puntos) Una matriz cuadrada es *equidiagonal* si el producto de los elementos de su diagonal principal es igual al producto de los elementos de su diagonal secundaria, y sus cuatro submatrices son a su vez equidiagonales. Se pide:

1. (1.75 puntos) Escribe un algoritmo recursivo *eficiente* que permita resolver el problema para una matriz dada suponiendo que el número de filas es una potencia de dos.
2. (0.75 puntos) Escribe la *recurrencia* que corresponde al coste de la función recursiva indicando claramente cuál es el tamaño del problema. Indica también a qué *orden de complejidad asintótica* pertenece dicho coste.

Entrada

La entrada comienza con una línea que contiene el número de casos de prueba. Cada caso de prueba contendrá el número de filas de la matriz y a continuación una línea con el contenido de cada una de las filas de la misma.

Salida

Por cada caso de prueba el programa escribirá `NO` si la matriz no es equidiagonal según la definición o `SI` seguido del valor del producto de la diagonal.

Entrada de ejemplo

```
5
1
3
2
1 2
2 4
2
1 2
3 4
4
1 2 -1 3
2 4 1 3
0 2 5 7
0 6 0 0
4
1 2 6 2
2 4 9 3
0 2 5 7
0 6 0 0
```

Salida de ejemplo

```
SI 3
SI 4
NO
NO
SI 0
```

3. (4 puntos) La Comunidad de FALacia tiene $m \geq 1$ maravillosas playas visitadas por miles de turistas que hay que mantener limpias. El jefe de servicio de limpieza de las playas dispone de un listado con los kilogramos de basura que hay en cada playa. Dispone de $n \geq 1$ personas para hacer la limpieza y tiene una tabla que le indica para cada persona, cuántos kilogramos de basura puede recoger en cada una de las playas (esto depende de muchos factores como la localización de la playa, su accesibilidad, la disponibilidad de la persona etc.) A una playa pueden enviarse cualquier número de personas pero cada persona solo puede enviarse a una playa. El objetivo es maximizar la cantidad de basura recogida teniendo en cuenta que el jefe de la Comunidad le ha ordenado que al menos l playas ($l \geq 0$, $l \leq n$ y $l \leq m$) han de estar completamente limpias (una playa estará completamente limpia cuando la cantidad de basura que puede ser recogida por las personas enviadas a esa playa es superior o igual a la cantidad de basura que hay en esa playa).

- (0.25 puntos) Define el *espacio de soluciones* e indica cómo es el *árbol de exploración*.
- (2.25 puntos) Implementa un algoritmo de *vuelta atrás* que resuelva el problema. Explica claramente los *marcadores* que has utilizado.
- (1.5 puntos) Plantea dos posibles funciones de poda de optimalidad, razona sobre cual de ellas es mejor e impleméntala en tu algoritmo.

Entrada

La entrada comienza con una línea que contiene el número de casos de prueba. Cada caso de prueba contendrá el valor del número de personas n, de playas m y el número de playas que obligatoriamente hay que dejar completamente limpias como mínimo l. A continuación figura la cantidad de kilos de basura en cada playa. Finalmente una fila para cada persona indicando la cantidad de basura que puede recoger en cada una de las playas.

Salida

Por cada caso de prueba el programa escribirá una línea con la cantidad de basura máxima que se puede recoger y el número de playas completamente limpias. En caso de no sea posible cumplir la restricción de limpiar completamente al menos l playas se escribirá `IMPOSIBLE`.

OBSERVA:

- En el primero de los ejemplos que hay a continuación, donde solo hay una playa, se deja dicha playa completamente limpia. El máximo recogido son los 3 kilos aunque que la persona enviada a dicha playa podría haber recogido más si hubiese.
- Los ejemplos segundo, tercero y cuarto tienen las mismas playas y personas. En el segundo ejemplo se recogen 13 kilos obtenidos al enviar a la playa 1 las personas 0 y 3, y a la playa 2 las personas 1 y 2: $13 = 3 + 3 + 3 + 4$. Esas dos playas quedan completamente limpias, mientras que no se envía a nadie a la playa 0. En el tercer ejemplo donde se exige dejar limpias todas las playas resulta IMPOSIBLE, mientras que en el cuarto ejemplo donde no se exige limpiar ninguna playa el máximo obtenido es mayor: $14 = 3 + 4 + 3 + 4$ y ninguna playa completamente limpia.
- Si hay una playa con basura 0, cuenta como playa completamente limpia.

Entrada de ejemplo

```
4
1 1 1
3
5
4 3 2
10 7 6
2 3 1
4 3 3
2 1 3
4 4 4
4 3 3
10 7 6
2 3 1
4 3 3
2 1 3
4 4 4
4 3 0
10 7 6
2 3 1
4 3 3
2 1 3
4 4 4
```

Salida de ejemplo

```
3 1
13 2
IMPOSIBLE
14 0
```